Date=19/11/2020
Lecture By=Manish Mahant
Subject ⇒States And Props 3

IN PREVIOUS LECTURE **(QUICK RECAP)** **Date-18/11/2020**	**In Today's Lecture (Overview)**
Re-Render Component In React… The setState() method Code The forceUpdate() method Code How to use React Memo and what memoization actually means Questions For Self-Practice / Assignment For The Day	React onChange Events What is the onChange Event Handler? Example: Add an onChange Handler to an Input Example: Pass an Input Value from the onChange Event in a React Component Example: Saving an Input Value Inside of State using onChange in a React Component Question For Self-Practice // CC For the Day

React onChange Events

The onChange event in React detects when the value of an input element changes. Let's dive into some common examples of how to use onChange in React.

What is the onChange Event Handler?

JavaScript allows us to listen to an input's change in value by providing the attribute *onchange*. React's version of the onchange event handler is the same, but camel-cased.

If you're using forms inside of a React component, it's a good idea to understand how the onChange event handler works with forms, state, and how you can pass the value to a function.

Example: Add an onChange Handler to an Input

Let’s begin with a simple, but extremely important example: how to add an onChange event handler to an input, checkbox, or select element in a React component:

```
...

  return (
    <input name="firstName" onChange={handleChange} />
  );

  …
```

The above code displays a single input field which, when typed in, passes its current value to the *handleChange* function. As mentioned before, JavaScript’s native onChange syntax is written in all lowercase, however we use camel-case for our event handler names in React.

Example: Pass an Input Value from the onChange Event in a React Component

Let’s expand on the previous example by declaring a new function inside of our React component for the onChange handler to pass a value to. We’ll call it handleChange, and have it log the input’s current value to the console:

```
import React from 'react';
function App() {
  function handleChange(e) {
    console.log(e.target.value);
  }
  return (
    <input name="firstName" onChange={handleChange} />
  );
}
export default App;
```

An onChange event handler returns a *Synthetic Event* object which contains useful meta data such as the target input’s id, name, and current value.

We can access the target input’s value inside of the handleChange by accessing *e.target.value*.

Therefore, to log the name of the input field, we can log *e.target.name*.

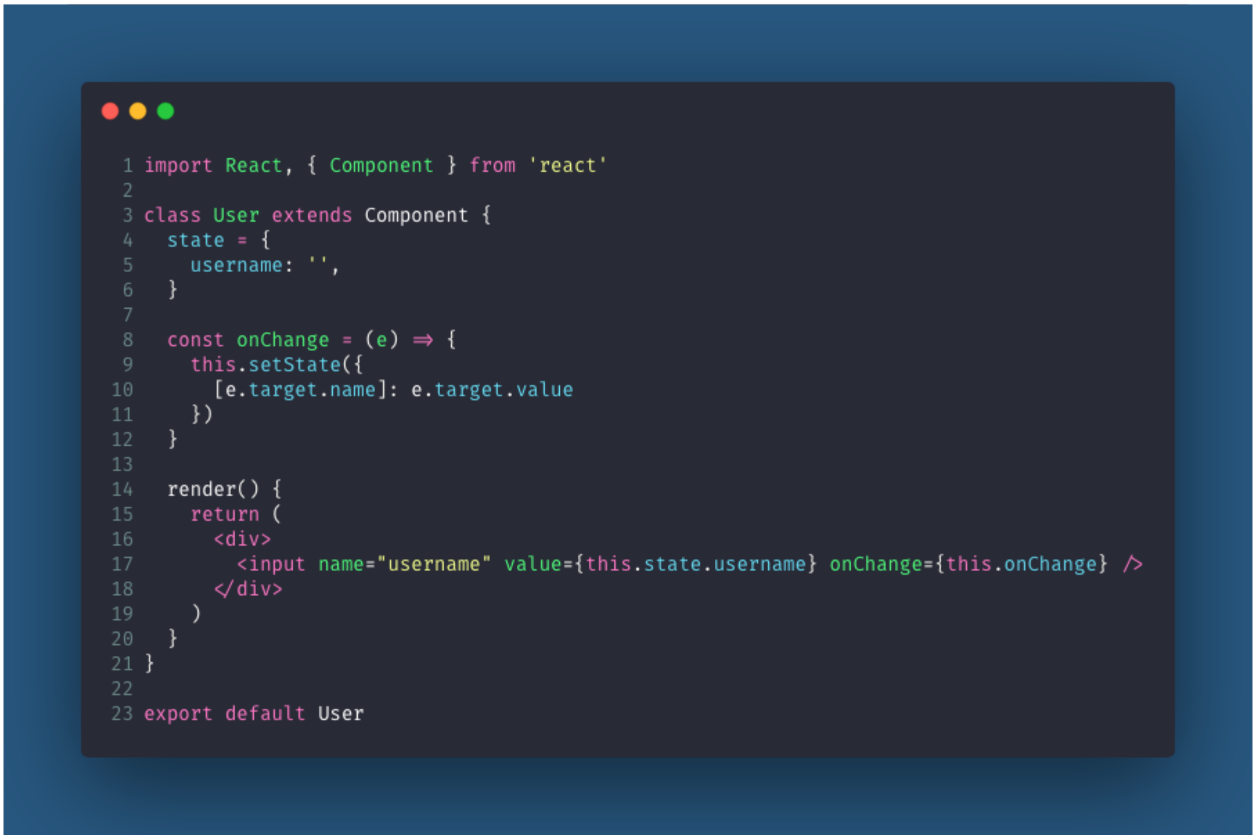
The example above was of a functional component. If you’re using a Class component, you will have to bind the onChange event handler to the context of this.

Example: Saving an Input Value Inside of State using onChange in a React Component

The final example we’ll explore today is how to store an input’s current value inside of a state value.

This is extremely useful for keeping track of the current input fields values, and displaying them on the UI.

```
import React from 'react';
function App() {
  const [firstName, setFirstName] = useState('');
  return (
    <input name="firstName" onChange={e => setFirstName(e.target.value)}
/>
  );
}
export default App;
```



```
import React, { Component } from 'react'

class User extends Component {
  state = {
    username: '',
  }

  const onChange = (e) => {
    this.setState({
      [e.target.name]: e.target.value
    })
  }

  render() {
    return (
      <div>
        <input name="username" value={this.state.username} onChange={this.onChange} />
      </div>
    )
  }
}

export default User
```

Question For Self-Practice // CC For the Day

https://au-assignment.s3.ap-south-1.amazonaws.com/Week_21_Day_4_Challenge-1aa76f75-5b4e-4ee9-b6de-9ea43e64df3e.pdf